MLP neural network (Fashion mnist)

Aria Hassanali Aragh . 99222032

In [1]:
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Dense, BatchNormalization
from tensorflow.keras.layers import Dropout
from keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
from keras.utils import plot_model
import numpy as np
import pandas as pd
```

Loading Datasets

In [2]:
```
fashion_mnist_test  = pd.read_csv("./datasets/fashion-mnist_test.csv")
fashion_mnist_train = pd.read_csv("./datasets/fashion-mnist_train.csv")
```

Dataset overview

Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Zalando intends Fashion-MNIST to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits.

In [3]:
```
print("trainig dataset : " +str(fashion_mnist_train.shape))
print("testing dataset : " +str(fashion_mnist_test.shape))
```

```
trainig dataset : (60000, 785)
testing dataset : (10000, 785)
```

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

Each row is a separate image

In [4]:
```
1 fashion_mnist_train.head(10)
```

Out[4]:

	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel77
0	2	0	0	0	0	0	0	0	0	0	...	0	
1	9	0	0	0	0	0	0	0	0	0	...	0	
2	6	0	0	0	0	0	0	0	5	0	...	0	
3	0	0	0	0	1	2	0	0	0	0	...	3	
4	3	0	0	0	0	0	0	0	0	0	...	0	
5	4	0	0	0	5	4	5	5	3	5	...	7	
6	4	0	0	0	0	0	0	0	0	0	...	14	
7	5	0	0	0	0	0	0	0	0	0	...	0	
8	4	0	0	0	0	0	0	3	2	0	...	1	
9	8	0	0	0	0	0	0	0	0	0	...	203	21

10 rows × 785 columns

Remaining columns are pixel numbers (784 total). Each value is the darkness of the pixel (1 to 255)

In [5]:
```
1 fashion_mnist_train.columns
```

Out[5]:
```
Index(['label', 'pixel1', 'pixel2', 'pixel3', 'pixel4', 'pixel5', 'pixel6',
       'pixel7', 'pixel8', 'pixel9',
       ...
       'pixel775', 'pixel776', 'pixel777', 'pixel778', 'pixel779', 'pixel78
0',
       'pixel781', 'pixel782', 'pixel783', 'pixel784'],
      dtype='object', length=785)
```

#Labels

Each training and test example is assigned to one of the following labels:

"0" T-shirt/top "1" Trouser "2" Pullover "3" Dress "4" Coat "5" Sandal "6" Shirt "7" Sneaker "8" Bag "9" Ankle boot

In [6]:
```
def label_to_string(argument):
    if argument == 0:
        return "T-shirt/top"
    elif argument == 1:
        return "Trouser"
    elif argument == 2:
         return "Pullover"
    elif argument == 3:
        return " Dress"
    elif argument == 4:
        return "Coat"
    elif argument == 5:
        return "Sandal"
    elif argument == 6:
        return "Shirt"
    elif argument == 7:
        return "Sneaker"
    elif argument == 8:
        return "Bag"
    elif argument == 9:
        return "Ankle boot"


```

In [7]:
```
fashion_mnist_train.loc[1, fashion_mnist_train.columns != "label"]
```

Out[7]:
```
pixel1      0
pixel2      0
pixel3      0
pixel4      0
pixel5      0
           ..
pixel780    0
pixel781    0
pixel782    0
pixel783    0
pixel784    0
Name: 1, Length: 784, dtype: int64
```

Checking function

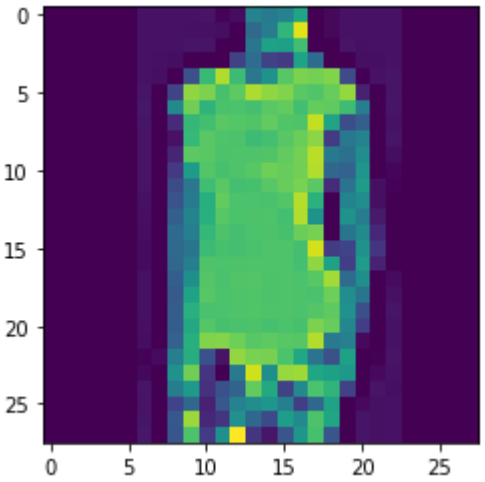
In [4]:
```
trainingAsMatrix = np.asmatrix(fashion_mnist_train)
```

In [73]:
```
def Check(a):
    label = fashion_mnist_train.loc[a][0]
    label = label_to_string(label)
    image = trainingAsMatrix[a,1:].reshape(28,28)
    print("image number " + str(a) )
    print("label : " + str(label))
    plt.imshow(image)
    plt.show

```

In [10]:
```
Check(80)
```

```
image number 80
label : T-shirt/top
```



In [6]:
```
X = trainingAsMatrix[:,1:]
```

In [8]:
```
X.shape
```

Out[8]: (60000, 784)

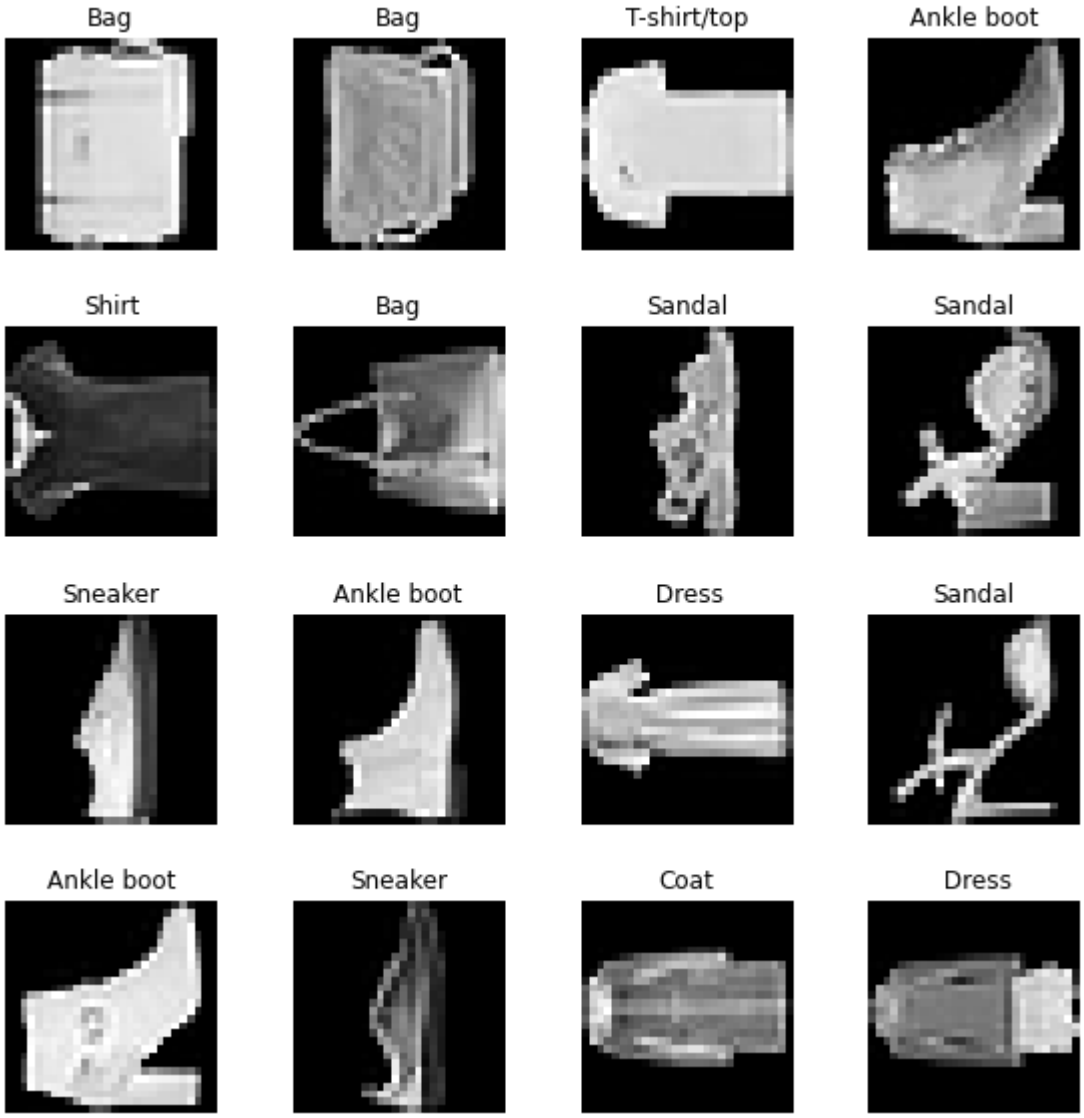
Plotting

In [74]:
```
Y = trainingAsMatrix[:,0]
```

In [75]:

```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
X = trainingAsMatrix[:,1:]
m, n = X.shape
fig, axes = plt.subplots(4,4, figsize=(8,8))
fig.tight_layout(pad=0.1)

for i,ax in enumerate(axes.flat):
    random_index = np.random.randint(m)
    X_random_reshaped = X[random_index].reshape((28,28)).T
    ax.imshow(X_random_reshaped, cmap='gray')
    ax.set_title(label_to_string(fashion_mnist_train.loc[random_index][0]))
    ax.set_axis_off()
```



Modeling

First we implement simple model using Tensorflow

In [76]:

```
model = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=56,activation=tf.nn.relu ,name="layer1"),
        Dense(units=128,activation=tf.nn.relu ,name="layer2"),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [17]:

```
model.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

model.fit(
    X,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 8s 4ms/step - loss: 1.3966 - cat
egorical_accuracy: 0.1132
Epoch 2/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.8021 - cat
egorical_accuracy: 0.0956
Epoch 3/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.7746 - cat
egorical_accuracy: 0.0926
Epoch 4/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.7613 - cat
egorical_accuracy: 0.0949
Epoch 5/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.7186 - cat
egorical_accuracy: 0.0984
Epoch 6/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.7064 - cat
egorical_accuracy: 0.1013
Epoch 7/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.6695 - cat
egorical_accuracy: 0.1045
Epoch 8/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.6298 - cat
egorical_accuracy: 0.1047
Epoch 9/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.6142 - cat
egorical_accuracy: 0.1069
Epoch 10/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.5961 - cat
egorical_accuracy: 0.1060
Epoch 11/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.5840 - cat
egorical_accuracy: 0.1060
Epoch 12/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5824 - cat
egorical_accuracy: 0.1064
Epoch 13/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5665 - cat
egorical_accuracy: 0.1070
Epoch 14/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.5694 - cat
egorical_accuracy: 0.1080
Epoch 15/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.5577 - cat
egorical_accuracy: 0.1046
Epoch 16/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5536 - cat
egorical_accuracy: 0.1016
Epoch 17/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5585 - cat
egorical_accuracy: 0.1031
Epoch 18/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5730 - cat
egorical_accuracy: 0.1096
Epoch 19/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5489 - cat
egorical_accuracy: 0.1042
```

```
Epoch 20/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.5295 - cat
egorical_accuracy: 0.1019
```

Out[17]: <keras.callbacks.History at 0x1fd90c93490>

Normalizing

In [77]:
```
1 normalizationX = X/255
```

In [19]:

```
model.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

model.fit(
    normalizationX,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.5482 - cat
egorical_accuracy: 0.1044
Epoch 2/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.4073 - cat
egorical_accuracy: 0.1056
Epoch 3/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3784 - cat
egorical_accuracy: 0.1054
Epoch 4/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3618 - cat
egorical_accuracy: 0.1048
Epoch 5/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.3485 - cat
egorical_accuracy: 0.1041
Epoch 6/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3391 - cat
egorical_accuracy: 0.1041
Epoch 7/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3321 - cat
egorical_accuracy: 0.1037
Epoch 8/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3253 - cat
egorical_accuracy: 0.1034
Epoch 9/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3188 - cat
egorical_accuracy: 0.1033
Epoch 10/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3117 - cat
egorical_accuracy: 0.1034
Epoch 11/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3067 - cat
egorical_accuracy: 0.1032
Epoch 12/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3009 - cat
egorical_accuracy: 0.1031
Epoch 13/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2965 - cat
egorical_accuracy: 0.1029
Epoch 14/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.2929 - cat
egorical_accuracy: 0.1037
Epoch 15/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2892 - cat
egorical_accuracy: 0.1027
Epoch 16/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.2856 - cat
egorical_accuracy: 0.1025
Epoch 17/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.2811 - cat
egorical_accuracy: 0.1025
Epoch 18/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2795 - cat
egorical_accuracy: 0.1033
Epoch 19/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2752 - cat
egorical_accuracy: 0.1021
```

```
Epoch 20/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.2709 - cat
egorical_accuracy: 0.1024
```

Out[19]: <keras.callbacks.History at 0x1fd90dc0be0>

Batch Normalization

Without normalizing

In [78]:

```
modelBatch = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=128,activation=tf.nn.relu ,name="layer1"),
        BatchNormalization(),
        Dense(units=512,activation=tf.nn.relu ,name="layer2"),
        BatchNormalization(),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [21]:

```
modelBatch.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelBatch.fit(
    X,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 11s 5ms/step - loss: 0.5153 - ca
tegorical_accuracy: 0.1023
Epoch 2/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.4052 - ca
tegorical_accuracy: 0.1023
Epoch 3/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3632 - ca
tegorical_accuracy: 0.1025
Epoch 4/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3409 - ca
tegorical_accuracy: 0.1024
Epoch 5/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3244 - ca
tegorical_accuracy: 0.1020
Epoch 6/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3088 - ca
tegorical_accuracy: 0.1015
Epoch 7/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2976 - ca
tegorical_accuracy: 0.1030
Epoch 8/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2846 - ca
tegorical_accuracy: 0.1015
Epoch 9/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2750 - ca
tegorical_accuracy: 0.1025
Epoch 10/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2626 - ca
tegorical_accuracy: 0.1020
Epoch 11/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2546 - ca
tegorical_accuracy: 0.1025
Epoch 12/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2494 - ca
tegorical_accuracy: 0.1016
Epoch 13/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2415 - ca
tegorical_accuracy: 0.1017
Epoch 14/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2364 - ca
tegorical_accuracy: 0.1019
Epoch 15/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2323 - ca
tegorical_accuracy: 0.1018
Epoch 16/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2253 - ca
tegorical_accuracy: 0.1012
Epoch 17/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2214 - ca
tegorical_accuracy: 0.1012
Epoch 18/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2161 - ca
tegorical_accuracy: 0.1019
Epoch 19/20
1875/1875 [==============================] - 10s 6ms/step - loss: 0.2161 - ca
tegorical_accuracy: 0.1013
```

```
Epoch 20/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2055 - ca
tegorical_accuracy: 0.1012
```

Out[21]: <keras.callbacks.History at 0x1fd910b1b50>

Add normalization

In [79]:

```
modelBatch.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelBatch.fit(
    normalizationX,Y,
    epochs=20
)
```

```
Epoch 1/20
 681/1875 [=========>....................] - ETA: 6s - loss: 0.5915 - categor
ical_accuracy: 0.1069
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_20516/1874524359.py in <module>
      5 )
      6
----> 7 modelBatch.fit(
      8     normalizationX,Y,
      9     epochs=20

~\anaconda3\lib\site-packages\keras\utils\traceback_utils.py in error_handler
(*args, **kwargs)
     62     filtered_tb = None
     63     try:
---> 64       return fn(*args, **kwargs)
     65     except Exception as e:  # pylint: disable=broad-except
     66       filtered_tb = _process_traceback_frames(e.__traceback__)

~\anaconda3\lib\site-packages\keras\engine\training.py in fit(self, x, y, bat
ch_size, epochs, verbose, callbacks, validation_split, validation_data, shuff
le, class_weight, sample_weight, initial_epoch, steps_per_epoch, validation_s
teps, validation_batch_size, validation_freq, max_queue_size, workers, use_mu
ltiprocessing)
   1407                 _r=1):
   1408               callbacks.on_train_batch_begin(step)
-> 1409               tmp_logs = self.train_function(iterator)
   1410               if data_handler.should_sync:
   1411                 context.async_wait()

~\anaconda3\lib\site-packages\tensorflow\python\util\traceback_utils.py in er
ror_handler(*args, **kwargs)
    148     filtered_tb = None
    149     try:
--> 150       return fn(*args, **kwargs)
    151     except Exception as e:
    152       filtered_tb = _process_traceback_frames(e.__traceback__)

~\anaconda3\lib\site-packages\tensorflow\python\eager\def_function.py in __ca
ll__(self, *args, **kwds)
    913
    914       with OptionalXlaContext(self._jit_compile):
--> 915         result = self._call(*args, **kwds)
    916
    917       new_tracing_count = self.experimental_get_tracing_count()

~\anaconda3\lib\site-packages\tensorflow\python\eager\def_function.py in _cal
l(self, *args, **kwds)
    945       # In this case we have created variables on the first call, so
we run the
    946       # defunned version which is guaranteed to never create variable
s.
--> 947       return self._stateless_fn(*args, **kwds)  # pylint: disable=not
-callable
    948     elif self._stateful_fn is not None:
    949       # Release the lock early so that multiple threads can perform t
he call

~\anaconda3\lib\site-packages\tensorflow\python\eager\function.py in __call__
```

```
(self, *args, **kwargs)
   2451       (graph_function,
   2452        filtered_flat_args) = self._maybe_define_function(args, kwarg
s)
-> 2453     return graph_function._call_flat(
   2454         filtered_flat_args, captured_inputs=graph_function.captured_i
nputs)  # pylint: disable=protected-access
   2455

~\anaconda3\lib\site-packages\tensorflow\python\eager\function.py in _call_fl
at(self, args, captured_inputs, cancellation_manager)
   1858         and executing_eagerly):
   1859       # No tape is watching; skip to running the function.
-> 1860       return self._build_call_outputs(self._inference_function.call(
   1861           ctx, args, cancellation_manager=cancellation_manager))
   1862     forward_backward = self._select_forward_and_backward_functions(

~\anaconda3\lib\site-packages\tensorflow\python\eager\function.py in call(sel
f, ctx, args, cancellation_manager)
    495       with _InterpolateFunctionError(self):
    496         if cancellation_manager is None:
--> 497           outputs = execute.execute(
    498               str(self.signature.name),
    499               num_outputs=self._num_outputs,

~\anaconda3\lib\site-packages\tensorflow\python\eager\execute.py in quick_exe
cute(op_name, num_outputs, inputs, attrs, ctx, name)
     52   try:
     53     ctx.ensure_initialized()
---> 54     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_
name,
     55                                         inputs, attrs, num_outputs)
     56   except core._NotOkStatusException as e:

KeyboardInterrupt:
```

Dropout

In [22]:
```
1 # 0.2
```

In [16]:

```
modelDropout_2 = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=128,activation=tf.nn.relu ,name="layer1"),
        Dropout(0.2),
        Dense(units=512,activation=tf.nn.relu ,name="layer2"),
        Dropout(0.2),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [24]:

```
modelDropout_2.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelDropout_2.fit(
    normalizationX,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.5366 - cat
egorical_accuracy: 0.1057
Epoch 2/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4146 - cat
egorical_accuracy: 0.1042
Epoch 3/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3853 - cat
egorical_accuracy: 0.1032
Epoch 4/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3643 - cat
egorical_accuracy: 0.1032
Epoch 5/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3495 - cat
egorical_accuracy: 0.1027
Epoch 6/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3377 - cat
egorical_accuracy: 0.1025
Epoch 7/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3262 - cat
egorical_accuracy: 0.1033
Epoch 8/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3183 - cat
egorical_accuracy: 0.1027
Epoch 9/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3131 - cat
egorical_accuracy: 0.1035
Epoch 10/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3052 - cat
egorical_accuracy: 0.1037
Epoch 11/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2991 - cat
egorical_accuracy: 0.1027
Epoch 12/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.2926 - cat
egorical_accuracy: 0.1028
Epoch 13/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2899 - cat
egorical_accuracy: 0.1029
Epoch 14/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.2828 - cat
egorical_accuracy: 0.1024
Epoch 15/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.2799 - cat
egorical_accuracy: 0.1018
Epoch 16/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.2772 - cat
egorical_accuracy: 0.1030
Epoch 17/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.2724 - cat
egorical_accuracy: 0.1020
Epoch 18/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2667 - cat
egorical_accuracy: 0.1020
Epoch 19/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2662 - cat
egorical_accuracy: 0.1025
```

```
Epoch 20/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2624 - cat
egorical_accuracy: 0.1025
```

Out[24]:
```
<keras.callbacks.History at 0x1fd914ceaf0>
```

In [36]:
```
#0.3
```

In [17]:
```
modelDropout_3 = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=128,activation=tf.nn.relu ,name="layer1"),
        Dropout(0.3),
        Dense(units=512,activation=tf.nn.relu ,name="layer2"),
        Dropout(0.3),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [26]:

```
modelDropout_3.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelDropout_3.fit(
    normalizationX,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 9s 4ms/step - loss: 0.5773 - cat
egorical_accuracy: 0.1067
Epoch 2/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4417 - cat
egorical_accuracy: 0.1042
Epoch 3/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.4119 - cat
egorical_accuracy: 0.1046
Epoch 4/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3909 - cat
egorical_accuracy: 0.1043
Epoch 5/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3779 - cat
egorical_accuracy: 0.1032
Epoch 6/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3669 - cat
egorical_accuracy: 0.1034
Epoch 7/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3576 - cat
egorical_accuracy: 0.1028
Epoch 8/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3507 - cat
egorical_accuracy: 0.1037
Epoch 9/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3483 - cat
egorical_accuracy: 0.1036
Epoch 10/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3401 - cat
egorical_accuracy: 0.1028
Epoch 11/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3295 - cat
egorical_accuracy: 0.1030
Epoch 12/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3313 - cat
egorical_accuracy: 0.1037
Epoch 13/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3221 - cat
egorical_accuracy: 0.1033
Epoch 14/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3194 - cat
egorical_accuracy: 0.1034
Epoch 15/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3160 - cat
egorical_accuracy: 0.1033
Epoch 16/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3159 - cat
egorical_accuracy: 0.1037
Epoch 17/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3088 - cat
egorical_accuracy: 0.1031
Epoch 18/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3087 - cat
egorical_accuracy: 0.1039
Epoch 19/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3046 - cat
egorical_accuracy: 0.1023
```

```
Epoch 20/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3028 - cat
egorical_accuracy: 0.1029
```

Out[26]:
```
<keras.callbacks.History at 0x1fd914bdeb0>
```

In [37]:
```
#0.4
```

In [18]:
```
modelDropout_4 = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=128,activation=tf.nn.relu ,name="layer1"),
        Dropout(0.4),
        Dense(units=512,activation=tf.nn.relu ,name="layer2"),
        Dropout(0.4),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [30]:

```
modelDropout_4.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelDropout_4.fit(
    normalizationX,Y,
    epochs=20
)
```

```
Epoch 1/20
1875/1875 [==============================] - 9s 4ms/step - loss: 0.6155 - cat
egorical_accuracy: 0.1075
Epoch 2/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4753 - cat
egorical_accuracy: 0.1037
Epoch 3/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.4448 - cat
egorical_accuracy: 0.1032
Epoch 4/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4226 - cat
egorical_accuracy: 0.1045
Epoch 5/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.4145 - cat
egorical_accuracy: 0.1039
Epoch 6/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.4009 - cat
egorical_accuracy: 0.1036
Epoch 7/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3933 - cat
egorical_accuracy: 0.1044
Epoch 8/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3846 - cat
egorical_accuracy: 0.1036
Epoch 9/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3810 - cat
egorical_accuracy: 0.1037
Epoch 10/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3777 - cat
egorical_accuracy: 0.1046
Epoch 11/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3655 - cat
egorical_accuracy: 0.1032
Epoch 12/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3644 - cat
egorical_accuracy: 0.1040
Epoch 13/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3601 - cat
egorical_accuracy: 0.1051
Epoch 14/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3582 - cat
egorical_accuracy: 0.1038
Epoch 15/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3523 - cat
egorical_accuracy: 0.1030
Epoch 16/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3532 - cat
egorical_accuracy: 0.1038
Epoch 17/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3508 - cat
egorical_accuracy: 0.1038
Epoch 18/20
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3512 - cat
egorical_accuracy: 0.1032
Epoch 19/20
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3452 - cat
egorical_accuracy: 0.1038
```

```
Epoch 20/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3400 - cat
egorical_accuracy: 0.1025
```

Out[30]: <keras.callbacks.History at 0x1fd912f4eb0>

Early stopping

In [80]:
```
1 ES = EarlyStopping(monitor='categorical_accuracy', patience=3)
```

In [81]:

```
modelBatch.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

modelBatch.fit(
    normalizationX,Y,
    epochs=20,callbacks=[ES]
)
```

```
Epoch 1/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.4503 - ca
tegorical_accuracy: 0.1034
Epoch 2/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3819 - cat
egorical_accuracy: 0.1024
Epoch 3/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.3568 - ca
tegorical_accuracy: 0.1030
Epoch 4/20
1579/1875 [========================>.....] - ETA: 1s - loss: 0.3301 - categor
ical_accuracy: 0.1027
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_20516/3205273382.py in <module>
      5 )
      6
----> 7 modelBatch.fit(
      8     normalizationX,Y,
      9     epochs=20,callbacks=[ES]

~\anaconda3\lib\site-packages\keras\utils\traceback_utils.py in error_handler
(*args, **kwargs)
     62     filtered_tb = None
     63     try:
---> 64       return fn(*args, **kwargs)
     65     except Exception as e:  # pylint: disable=broad-except
     66       filtered_tb = _process_traceback_frames(e.__traceback__)

~\anaconda3\lib\site-packages\keras\engine\training.py in fit(self, x, y, bat
ch_size, epochs, verbose, callbacks, validation_split, validation_data, shuff
le, class_weight, sample_weight, initial_epoch, steps_per_epoch, validation_s
teps, validation_batch_size, validation_freq, max_queue_size, workers, use_mu
ltiprocessing)
   1411                 context.async_wait()
   1412               logs = tmp_logs  # No error, now safe to assign to log
s.
-> 1413               end_step = step + data_handler.step_increment
   1414               callbacks.on_train_batch_end(end_step, logs)
   1415               if self.stop_training:

~\anaconda3\lib\site-packages\keras\engine\data_adapter.py in step_increment
(self)
   1266         self._steps_per_execution.assign(original_spe)
   1267
-> 1268   @property
   1269   def step_increment(self):
   1270     """The number to increment the step for `on_batch_end` method
s."""

KeyboardInterrupt:
```

L1 and L2

In [20]:

```
model = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=56,activation=tf.nn.relu ,kernel_regularizer='l1',name
        Dense(units=128,activation=tf.nn.relu ,kernel_regularizer='l1',nam
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")

    ], name = "simpleTensorflowModel"
)
```

In [43]:

```
model.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)

model.fit(
    normalizationX,Y,
    epochs=20,
)
```

```
Epoch 1/20
1875/1875 [==============================] - 7s 3ms/step - loss: 2.0622 - cat
egorical_accuracy: 0.1107
Epoch 2/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.1989 - cat
egorical_accuracy: 0.1102
Epoch 3/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.1273 - cat
egorical_accuracy: 0.1093
Epoch 4/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0883 - cat
egorical_accuracy: 0.1081
Epoch 5/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0640 - cat
egorical_accuracy: 0.1074
Epoch 6/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0455 - cat
egorical_accuracy: 0.1076
Epoch 7/20
1875/1875 [==============================] - 7s 4ms/step - loss: 1.0337 - cat
egorical_accuracy: 0.1064
Epoch 8/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0249 - cat
egorical_accuracy: 0.1068
Epoch 9/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0119 - cat
egorical_accuracy: 0.1067
Epoch 10/20
1875/1875 [==============================] - 6s 3ms/step - loss: 1.0013 - cat
egorical_accuracy: 0.1067
Epoch 11/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9949 - cat
egorical_accuracy: 0.1070
Epoch 12/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9920 - cat
egorical_accuracy: 0.1064
Epoch 13/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9820 - cat
egorical_accuracy: 0.1060
Epoch 14/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9768 - cat
egorical_accuracy: 0.1060
Epoch 15/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9747 - cat
egorical_accuracy: 0.1055
Epoch 16/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9662 - cat
egorical_accuracy: 0.1062
Epoch 17/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9634 - cat
egorical_accuracy: 0.1063
Epoch 18/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9589 - cat
egorical_accuracy: 0.1060
Epoch 19/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9579 - cat
egorical_accuracy: 0.1059
```

```
Epoch 20/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.9624 - cat
egorical_accuracy: 0.1059
```

Out[43]: <keras.callbacks.History at 0x1fd9239ce20>

The Final Model

In [91]:
```
FinalModel = Sequential(
    [
        tf.keras.Input(shape=(784,)),
        Dense(units=128,activation=tf.nn.relu ,name="layer0"),
        BatchNormalization(),
        Dense(units=256,activation=tf.nn.relu ,name="layer1"),
        BatchNormalization(),
        Dropout(0.3),
        Dense(units=512,activation=tf.nn.relu ,name="layer2"),
        BatchNormalization(),
        Dropout(0.5),
        Dense(units=10 ,activation=tf.nn.softmax ,name="layer3")


    ], name = "FinalModel"
)
```

In [92]:
```
FinalModel.compile(
    loss= "sparse_categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(0.001),
    metrics=[tf.keras.metrics.CategoricalAccuracy()]
)
```

In [94]:

```
FinalModel.fit(
    normalizationX,Y,
    epochs=20,
)
```

```
Epoch 1/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.6271 - ca
tegorical_accuracy: 0.1024
Epoch 2/20
1875/1875 [==============================] - 13s 7ms/step - loss: 0.4606 - ca
tegorical_accuracy: 0.1031
Epoch 3/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.4284 - ca
tegorical_accuracy: 0.1025
Epoch 4/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.4018 - ca
tegorical_accuracy: 0.1031
Epoch 5/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.3861 - ca
tegorical_accuracy: 0.1032
Epoch 6/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3723 - ca
tegorical_accuracy: 0.1039
Epoch 7/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.3585 - ca
tegorical_accuracy: 0.1032
Epoch 8/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3470 - ca
tegorical_accuracy: 0.1020
Epoch 9/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.3394 - ca
tegorical_accuracy: 0.1028
Epoch 10/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3271 - ca
tegorical_accuracy: 0.1022
Epoch 11/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.3212 - ca
tegorical_accuracy: 0.1030
Epoch 12/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3136 - ca
tegorical_accuracy: 0.1038
Epoch 13/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3070 - ca
tegorical_accuracy: 0.1021
Epoch 14/20
1875/1875 [==============================] - 12s 6ms/step - loss: 0.3011 - ca
tegorical_accuracy: 0.1021
Epoch 15/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2944 - ca
tegorical_accuracy: 0.1019
Epoch 16/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.2912 - ca
tegorical_accuracy: 0.1020
Epoch 17/20
1875/1875 [==============================] - 12s 7ms/step - loss: 0.2883 - ca
tegorical_accuracy: 0.1022
Epoch 18/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2844 - ca
tegorical_accuracy: 0.1022
Epoch 19/20
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2771 - cat
egorical_accuracy: 0.1025
```

```
Epoch 20/20
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2733 - ca
tegorical_accuracy: 0.1027
```

Out[94]:
```
<keras.callbacks.History at 0x2745d7a9e50>
```

In [97]:
```
testMatrix = np.asmatrix(fashion_mnist_test)
xTest = testMatrix[:,1:]
yTest = testMatrix[:,0]
```

In [98]:
```
xTest=xTest/255
yTest=yTest/255
```

In [100]:
```
1 history = FinalModel.fit(normalizationX, Y, epochs=30, batch_size=200, va
```

```
Epoch 1/30
298/300 [============================>.] - ETA: 0s - loss: 0.2222 - categoric
al_accuracy: 0.1022
```

```
---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_20516/1654139771.py in <module>
----> 1 history = FinalModel.fit(normalizationX, Y, epochs=30, batch_size=20
0, validation_data=(xTest, yTest), callbacks=[ES])

~\anaconda3\lib\site-packages\keras\utils\traceback_utils.py in error_handler
(*args, **kwargs)
     65     except Exception as e:  # pylint: disable=broad-except
     66       filtered_tb = _process_traceback_frames(e.__traceback__)
---> 67       raise e.with_traceback(filtered_tb) from None
     68     finally:
     69       del filtered_tb

~\anaconda3\lib\site-packages\tensorflow\python\eager\execute.py in quick_exe
cute(op_name, num_outputs, inputs, attrs, ctx, name)
     52   try:
     53     ctx.ensure_initialized()
---> 54     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_
name,
     55                                         inputs, attrs, num_outputs)
     56   except core._NotOkStatusException as e:

InvalidArgumentError: Graph execution error:

Detected at node 'sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWi
thLogits/SparseSoftmaxCrossEntropyWithLogits' defined at (most recent call la
st):
    File "C:\Users\ASUS02\anaconda3\lib\runpy.py", line 197, in _run_module_a
s_main
      return _run_code(code, main_globals, None,
    File "C:\Users\ASUS02\anaconda3\lib\runpy.py", line 87, in _run_code
      exec(code, run_globals)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel_launcher.py",
line 16, in <module>
      app.launch_new_instance()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\traitlets\config\applic
ation.py", line 846, in launch_instance
      app.start()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelapp.p
y", line 677, in start
      self.io_loop.start()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\tornado\platform\asynci
o.py", line 199, in start
      self.asyncio_loop.run_forever()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\base_events.py", line 596, in
run_forever
      self._run_once()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\base_events.py", line 1890, i
n _run_once
      handle._run()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\events.py", line 80, in _run
      self._context.run(self._callback, *self._args)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 457, in dispatch_queue
      await self.process_one()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
```

```
y", line 446, in process_one
      await dispatch(*args)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 353, in dispatch_shell
      await result
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 648, in execute_request
      reply_content = await reply_content
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\ipkernel.py",
line 353, in do_execute
      res = shell.run_cell(code, store_history=store_history, silent=silent)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\zmqshell.py",
line 533, in run_cell
      return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 2901, in run_cell
      result = self._run_cell(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 2947, in _run_cell
      return runner(coro)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\async_help
ers.py", line 68, in _pseudo_sync_runner
      coro.send(None)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3172, in run_cell_async
      has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3364, in run_ast_nodes
      if (await self.run_code(code, result,  async_=asy)):
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3444, in run_code
      exec(code_obj, self.user_global_ns, self.user_ns)
    File "C:\Users\ASUS02\AppData\Local\Temp/ipykernel_20516/4007401357.py",
line 1, in <module>
      FinalModel.evaluate(xTest, yTest)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\utils\traceback_u
tils.py", line 64, in error_handler
      return fn(*args, **kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1756, in evaluate
      tmp_logs = self.test_function(iterator)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1557, in test_function
      return step_function(self, iterator)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1546, in step_function
      outputs = model.distribute_strategy.run(run_step, args=(data,))
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1535, in run_step
      outputs = model.test_step(data)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1501, in test_step
      self.compute_loss(x, y, y_pred, sample_weight)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 948, in compute_loss
      return self.compiled_loss(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\compile_ut
```

```
ils.py", line 201, in __call__
      loss_value = loss_obj(y_t, y_p, sample_weight=sw)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
139, in __call__
      losses = call_fn(y_true, y_pred)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
243, in call
      return ag_fn(y_true, y_pred, **self._fn_kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
1860, in sparse_categorical_crossentropy
      return backend.sparse_categorical_crossentropy(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\backend.py", line
5238, in sparse_categorical_crossentropy
      res = tf.nn.sparse_softmax_cross_entropy_with_logits(
Node: 'sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/Sp
arseSoftmaxCrossEntropyWithLogits'
labels must be 1-D, but got shape [200,1]
         [[{{node sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWi
thLogits/SparseSoftmaxCrossEntropyWithLogits}}]] [Op:__inference_test_functio
n_384862]
```

In [99]:
```
1 FinalModel.evaluate(xTest, yTest)
```

```
---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_20516/4007401357.py in <module>
----> 1 FinalModel.evaluate(xTest, yTest)

~\anaconda3\lib\site-packages\keras\utils\traceback_utils.py in error_handler
(*args, **kwargs)
     65     except Exception as e:  # pylint: disable=broad-except
     66       filtered_tb = _process_traceback_frames(e.__traceback__)
---> 67       raise e.with_traceback(filtered_tb) from None
     68     finally:
     69       del filtered_tb

~\anaconda3\lib\site-packages\tensorflow\python\eager\execute.py in quick_exe
cute(op_name, num_outputs, inputs, attrs, ctx, name)
     52   try:
     53     ctx.ensure_initialized()
---> 54     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_
name,
     55                                         inputs, attrs, num_outputs)
     56   except core._NotOkStatusException as e:

InvalidArgumentError: Graph execution error:

Detected at node 'sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWi
thLogits/SparseSoftmaxCrossEntropyWithLogits' defined at (most recent call la
st):
    File "C:\Users\ASUS02\anaconda3\lib\runpy.py", line 197, in _run_module_a
s_main
      return _run_code(code, main_globals, None,
    File "C:\Users\ASUS02\anaconda3\lib\runpy.py", line 87, in _run_code
      exec(code, run_globals)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel_launcher.py",
line 16, in <module>
      app.launch_new_instance()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\traitlets\config\applic
ation.py", line 846, in launch_instance
      app.start()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelapp.p
y", line 677, in start
      self.io_loop.start()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\tornado\platform\asynci
o.py", line 199, in start
      self.asyncio_loop.run_forever()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\base_events.py", line 596, in
run_forever
      self._run_once()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\base_events.py", line 1890, i
n _run_once
      handle._run()
    File "C:\Users\ASUS02\anaconda3\lib\asyncio\events.py", line 80, in _run
      self._context.run(self._callback, *self._args)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 457, in dispatch_queue
      await self.process_one()
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 446, in process_one
```

```
      await dispatch(*args)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 353, in dispatch_shell
      await result
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\kernelbase.p
y", line 648, in execute_request
      reply_content = await reply_content
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\ipkernel.py",
line 353, in do_execute
      res = shell.run_cell(code, store_history=store_history, silent=silent)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\ipykernel\zmqshell.py",
line 533, in run_cell
      return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 2901, in run_cell
      result = self._run_cell(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 2947, in _run_cell
      return runner(coro)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\async_help
ers.py", line 68, in _pseudo_sync_runner
      coro.send(None)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3172, in run_cell_async
      has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3364, in run_ast_nodes
      if (await self.run_code(code, result,  async_=asy)):
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\IPython\core\interactiv
eshell.py", line 3444, in run_code
      exec(code_obj, self.user_global_ns, self.user_ns)
    File "C:\Users\ASUS02\AppData\Local\Temp/ipykernel_20516/4007401357.py",
line 1, in <module>
      FinalModel.evaluate(xTest, yTest)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\utils\traceback_u
tils.py", line 64, in error_handler
      return fn(*args, **kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1756, in evaluate
      tmp_logs = self.test_function(iterator)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1557, in test_function
      return step_function(self, iterator)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1546, in step_function
      outputs = model.distribute_strategy.run(run_step, args=(data,))
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1535, in run_step
      outputs = model.test_step(data)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 1501, in test_step
      self.compute_loss(x, y, y_pred, sample_weight)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\training.p
y", line 948, in compute_loss
      return self.compiled_loss(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\engine\compile_ut
ils.py", line 201, in __call__
```

```
      loss_value = loss_obj(y_t, y_p, sample_weight=sw)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
139, in __call__
      losses = call_fn(y_true, y_pred)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
243, in call
      return ag_fn(y_true, y_pred, **self._fn_kwargs)
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\losses.py", line
1860, in sparse_categorical_crossentropy
      return backend.sparse_categorical_crossentropy(
    File "C:\Users\ASUS02\anaconda3\lib\site-packages\keras\backend.py", line
5238, in sparse_categorical_crossentropy
      res = tf.nn.sparse_softmax_cross_entropy_with_logits(
Node: 'sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/Sp
arseSoftmaxCrossEntropyWithLogits'
labels must be 1-D, but got shape [32,1]
         [[{{node sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWi
thLogits/SparseSoftmaxCrossEntropyWithLogits}}]] [Op:__inference_test_functio
n_384862]
```

In [87]:
```
1 FinalModel.predict(X_test)
```

```
313/313 [==============================] - 1s 4ms/step
```

Out[87]:
```
array([[5.5424297e-01, 7.3404670e-05, 1.0673282e-02, ..., 3.1928092e-05,
        5.8403326e-04, 3.9307517e-05],
       [1.1315753e-09, 1.0000000e+00, 5.0364229e-10, ..., 3.3443687e-10,
        2.7206801e-10, 7.4383083e-11],
       [1.2308758e-02, 3.1286944e-04, 7.0267153e-01, ..., 3.3505290e-04,
        1.0749992e-03, 1.2182353e-04],
       ...,
       [1.3365862e-08, 8.9979149e-11, 1.3401441e-09, ..., 7.7222373e-10,
        9.9999988e-01, 7.7830672e-11],
       [2.8001359e-02, 4.7363999e-04, 9.0498086e-03, ..., 2.0502871e-04,
        6.6361213e-01, 1.5982843e-04],
       [1.1719097e-02, 6.1053854e-01, 1.0452826e-01, ..., 5.7518261e-04,
        7.0165377e-03, 6.5083749e-04]], dtype=float32)
```

The summary

In [60]:
```
1 FinalModel.summary()
```

```
Model: "FinalModel"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 layer0 (Dense)              (None, 128)               100480

 batch_normalization_10 (Bat  (None, 128)              512
 chNormalization)

 layer1 (Dense)              (None, 256)               33024

 batch_normalization_11 (Bat  (None, 256)              1024
 chNormalization)

 dropout_14 (Dropout)        (None, 256)               0

 layer2 (Dense)              (None, 512)               131584

 batch_normalization_12 (Bat  (None, 512)              2048
 chNormalization)

 dropout_15 (Dropout)        (None, 512)               0

 layer3 (Dense)              (None, 10)                5130

=================================================================
Total params: 273,802
Trainable params: 272,010
Non-trainable params: 1,792
_________________________________________________________________
```

In [ ]:
```
1
```